Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Омский государственный технический университет»
Кафедра «Автоматизация системы обработки информации
и управления»

Отчет

по дисциплине:

«Современные инструментальные средства разработки программного обеспечения»

Выполнил:
студент группы ПИН-201
Салыкин А.И.
Подпись: _____
Проверил:
ст. преподаватель Кабанов
А.А.
Подпись: _____

Омск 2023

Содержание

Автоматизированная система управления учетными записями к информационным система в АО «”ГК” Титан»

Описание

Тема дипломной работы "Автоматизированная система управления учетными записями к информационным системам" является актуальной и востребованной в современном информационном обществе. Системы управления учетными записями (Identity and Access Management, IAM) играют важную роль в обеспечении безопасности и эффективности работы информационных систем в компаниях и учреждениях.

Целью данной работы является разработка и внедрение автоматизированной системы управления учетными записями, которая обеспечит централизованное управление доступом к информационным ресурсам организации. Такая система позволит эффективно управлять учетными записями пользователей и их правами доступа к различным приложениям, базам данных и другим информационным ресурсам.

Разработка автоматизированной системы управления учетными записями предполагает проведение следующих этапов работы:

1. Анализ требований к системе: изучение бизнес-процессов организации, определение требований к управлению учетными записями, анализ существующих решений на рынке.

2. Проектирование системы: разработка архитектуры системы, определение функциональных и нефункциональных требований, разработка схемы управления доступом и политик безопасности.

3. Разработка и тестирование: создание программного обеспечения для автоматизации процессов управления учетными записями, тестирование работы системы на тестовых данных.

4. Внедрение и настройка системы: установка и настройка системы на серверах организации, интеграция с существующими информационными системами, обучение пользователей работе с системой.

5. Эксплуатация и поддержка системы: обеспечение бесперебойной работы системы, регулярное обновление и исправление ошибок, поддержка пользователей и администраторов системы.

Результатом дипломной работы должна стать отработанная и функционирующая автоматизированная система управления учетными записями, способная эффективно обеспечивать безопасность и удобство доступа к информационным ресурсам организации. Такая система сможет существенно повысить производительность и эффективность работы сотрудников, а также упростить управление правами доступа и учетными записями в организации.

Стек технологий

Java: Основной язык программирования для разработки системы.

Spring Boot: Фреймворк для быстрой разработки и развертывания приложения.

Spring Security: Библиотека для обеспечения безопасности и контроля доступа к данным.

PostgreSQL: СУБД для хранения и управления данными об учетных записях.

Hibernate: ORM для взаимодействия с базой данных.

Thymeleaf: Библиотека шаблонов для генерации HTML-страниц.

Bootstrap: CSS фреймворк для стилизации веб-интерфейса.

Maven: Инструмент для управления зависимостями и сборки проекта.

Docker: Технология для контейнеризации и развертывания системы.

Jenkins: Система непрерывной интеграции для автоматизации процесса сборки и тестирования.

Для начала разработки необходимо выполнить следующие шаги:

Изучение и освоение выбранных технологий (Java, Spring Boot, Spring Security, Hibernate, Thymeleaf, Bootstrap, Maven, Docker, Jenkins).

Создание базы данных PostgreSQL для хранения информации об учетных записях информационных систем.

Разработка архитектуры приложения, определение основных компонентов и их взаимодействия.

Spring Boot — это фреймворк на основе Java, который упрощает разработку Spring-приложений. Он предоставляет инструменты для автоматической конфигурации, облегчения разработки RESTful API и других функций.

PostgreSQL — это система управления реляционными базами данных с открытым исходным кодом. Она используется для хранения данных и предоставления доступа к ним.

Hibernate — это ORM-библиотека, которая помогает разработчикам взаимодействовать с базами данных. Она предоставляет способ работы с данными без необходимости писать SQL-запросы вручную.

Thymeleaf — это шаблонизатор для Java, который позволяет создавать HTML, XHTML и XML-страницы. Он использует шаблоны на основе стандартных тегов и выражений, которые могут быть заменены динамическими данными.

Bootstrap — это CSS-фреймворк, который помогает создавать адаптивные веб-страницы с использованием готовых компонентов, таких как кнопки, формы, таблицы и т.д.

Maven — это инструмент для автоматизации сборки, тестирования и развертывания Java-проектов.

Автоматизированная система управления учетными записями (Identity and Access Management, IAM) является важной частью информационной системы любой организации. Она отвечает за управление доступом пользователей к различным ресурсам и информации внутри системы.

IAM-системы предоставляют средства для создания, управления и удаления учетных записей пользователей, а также управления их правами доступа. Они обеспечивают безопасность информационных систем, контролируя доступ к данным и ресурсам на основе политик безопасности и ролей пользователей.

Важные функции автоматизированной системы управления учетными записями включают:

1. Централизованное управление: IAM-системы предоставляют централизованный механизм для создания, изменения и удаления учетных записей пользователей. Они позволяют администраторам эффективно управлять доступом пользователей и поддерживать актуальность информации о пользователях.

2. Аутентификация и авторизация: IAM-системы обеспечивают механизмы аутентификации пользователей, проверяя их идентичность, например, с помощью пароля или двухфакторной аутентификации. Они также контролируют доступ пользователей к ресурсам системы на основе их прав доступа и политик безопасности.

3. Управление ролями и правами доступа: IAM-системы предоставляют возможность определения ролей пользователей и назначения им соответствующих прав доступа. Это позволяет упростить процесс управления доступом и обеспечить соответствие политик безопасности.

4. Учет и аудит: IAM-системы ведут учет действий пользователей, включая аудит доступа к ресурсам системы. Это позволяет обнаруживать и расследовать возможные инциденты безопасности, а также обеспечивает соответствие требованиям регуляторов.

5. Интеграция с другими системами: IAM-системы могут интегрироваться с другими информационными системами организации, такими как системы управления ресурсами или системы управления идентификацией, для обеспечения единообразия и эффективности процессов управления доступом.

Внедрение автоматизированной системы управления учетными записями позволяет организации повысить безопасность своих информационных систем, упростить процессы управления доступом и обеспечить соответствие требованиям регуляторов. Она также помогает

снизить риски, связанные с несанкционированным доступом и утечкой данных.

Описание дизайна

Дизайн приложения для автоматизированной системы управления учетными записями к информационным системам может быть организован следующим образом:

1. Главный экран:

- Отображается логотип компании и заголовок приложения.

- В верхней части расположена панель навигации, содержащая список информационных систем компании.

- Ниже панели навигации располагается блок информации о текущем пользователе, в данном случае о системном администраторе.

2. Раздел "Список сотрудников":

- На данной странице системный администратор может видеть список всех сотрудников компании.

- Каждый сотрудник представлен в виде плитки или строки, содержащей его основную информацию, такую как имя, фамилия и должность.

- Возможно добавление дополнительной информации о сотруднике, такой как контактная информация или номер учетной записи.

3. Раздел "Управление доступом":

- На этой странице системный администратор может просматривать и управлять доступом каждого сотрудника к информационным системам компании.

- Для каждой информационной системы будет предоставлена возможность выбора уровня доступа сотрудника, например, "Полный доступ", "Только чтение" или "Без доступа".

- Можно дополнительно предусмотреть фильтры и поисковую строку, чтобы упростить процесс нахождения и управления конкретным сотрудником.

4. Создание, блокировка и удаление учетных записей:

- На отдельной странице системный администратор имеет возможность создавать новые учетные записи для сотрудников, заполняя необходимую информацию, такую как имя, фамилия и должность.

- Возможность блокировки или удаления существующих учетных записей тоже должна быть предусмотрена, чтобы системный администратор мог контролировать доступ к информационным системам.

5. Раздел "Журнал действий":

- В данном разделе системный администратор может просмотреть историю всех действий, связанных с управлением учетными записями сотрудников и доступом к информационным системам.

- Журнал может содержать информацию о создании, изменении, блокировке или удалении учетных записей, а также о выдаче/изменении доступа к информационным системам.

Общий дизайн приложения должен быть интуитивно понятным и удобным в использовании, с четкой структурой и навигацией для ускорения процесса управления учетными записями сотрудников и доступом к информационным системам компании.

Примеры кодов

Одним из примеров IAM системы на Java может быть Apache Shiro. Shiro является мощным и гибким инструментом для аутентификации, авторизации и управления доступом в Java-приложениях.

Пример кода, демонстрирующий создание IAM системы с использованием Apache Shiro, может выглядеть следующим образом:

```
import org.apache.shiro.SecurityUtils;
import org.apache.shiro.authc.*;
import org.apache.shiro.config.IniSecurityManagerFactory;
import org.apache.shiro.mgt.SecurityManager;
import org.apache.shiro.subject.Subject;
import org.apache.shiro.util.Factory;
```

```
public class IAMSystem {

    public static void main(String[] args) {

        // 1. Создание фабрики для загрузки конфигурации Shiro
        Factory<SecurityManager> factory = new IniSecurityManagerFactory("classpath:shiro.ini");

        // 2. Создание SecurityManager с помощью фабрики
        SecurityManager securityManager = factory.getInstance();

        // 3. Установка SecurityManager в качестве глобальной инстанции
        SecurityUtils.setSecurityManager(securityManager);

        // 4. Получение текущего пользователя (Subject)
        Subject currentUser = SecurityUtils.getSubject();

        // 5. Создание токена для аутентификации пользователя
        UsernamePasswordToken token = new UsernamePasswordToken("username", "password");

        try {
            // 6. Попытка аутентификации пользователя
            currentUser.login(token);
            System.out.println("Пользователь успешно аутентифицирован!");

            // 7. Проверка наличия у пользователя определенной роли
            if (currentUser.hasRole("admin")) {
```

```
            System.out.println("Пользователь является администратором!");
        } else {
            System.out.println("Пользователь не является администратором!");
        }

        // 8. Проверка наличия у пользователя определенного разрешения
        if (currentUser.isPermitted("read:documents")) {
            System.out.println("Пользователь имеет доступ на чтение документов!");
        } else {
            System.out.println("Пользователь не имеет доступа на чтение документов!");
        }

        // 9. Выход пользователя из системы
        currentUser.logout();
        System.out.println("Пользователь успешно вышел из системы!");

    } catch (UnknownAccountException ex) {
        System.out.println("Неверное имя пользователя!");
    } catch (IncorrectCredentialsException ex) {
        System.out.println("Неверный пароль!");
    } catch (LockedAccountException ex) {
        System.out.println("Аккаунт заблокирован!");
    } catch (AuthenticationException ex) {
        System.out.println("Ошибка аутентификации!");
    }
```

```
    }
}
```

В данном примере Shiro используется для аутентификации пользователя с заданными именем пользователя и паролем, проверки его роли и доступа к ресурсам системы. Конфигурация Shiro определена в файле shiro.ini.

Пример IAM (Identity and Access Management) системы на Java Spring может выглядеть следующим образом:

Создание модели пользователей:

```
@Entity
@Table(name="users")
public class User {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private Long id;

    @Column(nullable=false, unique=true)
    private String username;

    @Column(nullable=false)
    private String password;

    // геттеры и сеттеры
}
```

Создание репозитория для работы с пользователями:

```
@Repository
public interface UserRepository extends JpaRepository<User, Long> {
    User findByUsername(String username);
}
```

Создание сервиса для управления пользователями:

```
@Service
public class UserService {
    @Autowired
    private UserRepository userRepository;

    public User createUser(String username, String password) {
        User user = new User();
        user.setUsername(username);
        user.setPassword(password);
        return userRepository.save(user);
    }

    public User findByUsername(String username) {
        return userRepository.findByUsername(username);
    }
}
```

Создание контроллера для обработки запросов от клиентов:

```
@RestController
public class UserController {
    @Autowired
    private UserService userService;

    @PostMapping("/register")
    public User registerUser(@RequestBody UserDto userDto) {
        // валидация данных пользователя и обработка ошибок

        User user = userService.createUser(userDto.getUsername(),
userDto.getPassword());
        // возвращение созданного пользователя
        return user;
```

```
    }

    @PostMapping("/login")
    public User loginUser(@RequestBody UserDto userDto) {
        // валидация данных пользователя и обработка ошибок

        User user = userService.findByUsername(userDto.getUsername());
        if (user != null && user.getPassword().equals(userDto.getPassword()))
{
            // авторизация пользователя
            return user;
        } else {
            // обработка ошибки входа
            return null;
        }
    }
}
```

Создание класса для передачи данных о пользователе между клиентом и сервером:

```
public class UserDto {
    private String username;
    private String password;

    // геттеры и сеттеры
}
```

Это только базовый пример и IAM системы обычно имеют более сложную структуру и много других функций, таких как управление правами доступа, управление сеансами пользователей и др. Но данный пример демонстрирует основные компоненты системы на Java Spring.

Схема для проекта "Автоматизированная система управления учетными записями к информационным системам" может включать следующие основные компоненты:

Пользователи ИС:

- Обычные пользователи ИС.
- Администраторы ИС.
- Менеджеры учетных записей.
- Встроенные учетные записи (например, системные аккаунты).

Учетные записи:

- Идентификационные данные (имя пользователя, пароль, почта).
- Роли и привилегии (администратор, пользователь, гость).
- Доступ к информационным ресурсам (файлы, базы данных, приложения).
- Журналы входа и активности (логи).

Центр управления учетными записями:

- Менеджер учетных записей - компонент, обеспечивающий функции управления учетными записями, такие как создание, изменение, удаление, блокировка и разблокировка аккаунтов.
- Интерфейс пользователя для управления учетными записями.

Взаимодействие с ИС:

- Интеграция с информационными системами - для обеспечения доступа к информации и ресурсам ИС.
- Механизм аутентификации и авторизации - для проверки и управления доступом пользователей к ИС.
- Шифрование данных - для обеспечения безопасности передаваемой информации.

База данных:

- Хранение учетных записей пользователей ИС.
- Хранение информации о правах доступа к ресурсам ИС.

- Хранение журналов активности пользователей.

Безопасность:

- Механизмы безопасности (шифрование, контроль доступа и т. д.).
- Антивирусное программное обеспечение - для защиты от вредоносных программ.
- Физические меры безопасности (закрытые серверные помещения, бэкапы данных и т. д.).

Это некоторые из основных компонентов и взаимодействий, которые могут быть присутствующими в схеме автоматизированной системы управления учетными записями к информационным системам. Зависимости и детали системы могут различаться в каждом конкретном проекте.